In the Claims

This listing of claims will replace all prior versions, and listings, or claims in the application:

1. (Currently Amended) A software engine for application loading a software application onto a user's machine, wherein a core service of the application is loaded along with the software engine onto the user's machine to enable the application to commence to operate on the user's machine, the engine subsequently loading non-core services of the application according to a priority order determined by the engine, wherein a non-core service is responsible for providing a functionality of the application, the engine uniquely determining the priority order for loading the non-core services at run time responsive to a user interaction during each execution of the software application and to an order with which the non-core services registered with the core service, wherein a download of a first non-core service ~~is responsible for providing a functionality of the application~~ may be interrupted to begin download of a second non-core service responsive to the user interaction.

2. (Original) A software engine as claimed in claim 1, wherein the engine is part of the core service and is loaded with the core service.

3. (Original) A software engine as claimed in claim 2, wherein the engine commences operation upon completion of loading of the core service.

4-6. (Canceled)

7. (Previously Presented) A software engine as claimed in claim 1, wherein before loading the non-core services they are registered with the engine.

8. (Previously Presented) A software engine as claimed in claim 7, wherein the engine checks a registration list of non-core services before loading a requested non-core service.

9. (Original) A software engine as claimed in claim 1, wherein there is provided a cache into which at least one object for the application can be stored.

10. (Original) A software engine as claimed in claim 9, wherein the engine includes a memory management module that keeps track of usage of cached objects; the memory management module being able to de-allocate one or more of the objects.

11. (Original) A software engine as claimed in claim 10, wherein the cache is operative only when the application is on the user's machine.

12. (Original) A software engine as claimed in claim 9, wherein the cache includes an object repository into which the at least one object is placed, and an object description.

13. (Original) A software engine as claimed in claim 12, wherein the object description includes one or more selected from the group consisting of: object reference, object key, reference counter and time stamp.

14. (Original) A software engine as claimed in claim 10, wherein the de-allocation of one or more of the objects includes an arbitrary time offset.

15. (Original) A software engine as claimed in claim 14, wherein if the object description of an object in the object repository has a reference counter equal to zero for a time equal to at least the time offset, the corresponding object description is removed from the object repository.

16. (Original) A software engine as claimed in any one of claim 1, wherein the loading is downloading over the Internet.

17. (Currently Amended) A method of loading a software application onto a user's machine using a software engine, the method including loading onto the user's machine core services of the

software application to enable the user to interact with the application along with a software engine for controlling the loading of non-core services of the software application, and loading non-core services of the application according to a priority order determined by the software engine, the software engine uniquely determining the priority order for loading the non-core services at run time responsive to a user interaction during each execution of the software application and to an order with which the non-core services register with the core service, ~~wherein a non-core service is responsible for providing a functionality of the application[,]~~ wherein a non-core service is responsible for providing a functionality of the application, wherein a download of a first non-core service may be interrupted to begin download of a second non-core service responsive to the user interaction.

18. (Original) A method as claimed in claim 17, wherein the engine is part of the core service and is loaded with the core service.

19. (Original) A method as claimed in claim 18, wherein the engine commences operation upon completion of loading of the core service.

20. (Original) A method as claimed in claim 17, wherein before loading the non-core services they are registered with the engine.

21. (Canceled)

22. (Previously Presented) A method as claimed in claim 17, wherein upon interaction with the application by the user, the application requests the engine to load at least one of the non-core services, and the engine checks a registration and gives the at least one non-core service top priority for loading.

23. (Canceled)

24. (Canceled)

25. (Original) A method as claimed in claim 17, wherein objects of the application are storable in a cache for reuse.

26. (Original) A method as claimed in claim 25, wherein cached objects are tracked using a memory management module of the engine, which can de-allocate one or more of the objects.

27. (Original) A method as claimed in claim 26, wherein the cache is operative only when the application is on the user's machine.

28. (Original) A method as claimed in claim 25, wherein the objects are placed into an object repository in the cache, together with an object description.

29. (Original) A method as claimed in claim 28, wherein the object description includes one or more selected from the group consisting of: object reference, object key, reference counter and time stamp.

30. (Original) A method as claimed in claim 26, wherein de-allocation includes an arbitrary time offset.

31. (Original) A method as claimed in claim 30, wherein if the object description of an object repository has a reference counter to equal to zero for a time equal to at least the time offset, the corresponding object description is removed from the object repository.

32. (Original) A method as claimed in claim 17, wherein the loading in downloading over the Internet.

33. (Previously Presented) A software engine as claimed in claim 1, further comprising a computer memory management system including a cache, and wherein objects of the application are storable in the cache for reuse.

34. (Previously Presented) A software engine as claimed in claim 33, wherein the cache is operative only when the application is on the user's machine.

35. (Previously Presented) A software engine as claimed in claim 33, wherein the objects are placed into an object repository in the cache, together with an object description.

36. (Previously Presented) A software engine as claimed in claim 35, wherein the object description includes one or more selected from the group consisting of: object reference, object key, reference counter and time stamp.

37. (Previously Presented) A software engine as claimed in claim 33, wherein cached objects are tracked using a memory management module, which can de-allocate one or more of the objects.

38. (Previously Presented) A software engine as claimed in claim 37, wherein de-allocation includes an arbitrary time offset.

39. (Previously Presented) A software engine as claimed in claim 38, wherein if the object description of an object repository has a reference counter to equal to zero for a time equal to at least the time offset, the corresponding object description is removed from the object repository.

40. (Previously Presented) A software engine as claimed in Claim 1, wherein the application comprises a non-browser application.

41. (Canceled)

42. (New) A software application, comprising:

at least one core service to enable the software application to operate on a computer;

a software engine, downloaded onto the computer with the at least one core service, for loading the software application onto the computer;

a plurality of non-core services, each of the plurality of non-core services providing a functionality of the software application, wherein the non-core services of the software application are downloaded onto the computer according to a priority order determined by the software engine, the software engine uniquely determining the priority order for loading the non-core services at run time of the software application responsive to a user interaction during each execution of the software application and to an order with which the non-core services register with the core service;

a object repository for storing previously used data objects and associating the previously used data objects with a unique key; and

wherein when execution of one of the non-core services requires the generation of one of the previously used data objects, the previously used data object is retrieved from the object repository rather than being generated by one of the plurality of non-core services using the unique key.